Fuzzy-logic system solves control problem

David I Brubaker, The Huntington Group, and Cedric Sheerer, C/S Associates

Complex, nonlinear control problems can yield to simple fuzzy-logic techniques that require no modeling. Although fuzzy logic does not produce an analytic solution, you can verify the solution's validity using simulation.

You can use fuzzy logic to control a complex system. The practical steps in the following sample problem show how to design a simple fuzzy-logic system, in this case, a system that controls a traffic light for a freeway's on ramp. The traffic light will automatically control the traffic flow onto a freeway (Fig 1). The design goal is to minimize the impact of the inflow traffic on the prevailing freeway traffic.

Currently, on northern California freeways, a red-and-green traffic light meters on-ramp traffic. The traffic light has a constant green-to-red cycle. An accompanying sign states that a single vehicle is allowed onto the freeway for each green light. A fixed timer switches metering on automatically during periods that usually have heavy traffic—typically during commuting times.

For our design, we would like to add a bit of complexity—and hopefully capability—to the system. First, we will allow the delay between green lights to depend on the prevailing freeway traffic's speed and density: The delay increases for higher traffic density and decreases for higher freeway speed. And second, by controlling the length of time the green light is active, we will allow potentially more than one car onto the freeway during each green light. A standard traffic light, having green, yellow, and red lights, will perform the metering.

The system will be fuzzy; that is, it will continuously monitor the inputs and from them determine appropriate output responses based on rules stated in words—or, more formally, linguistically stated control criteria. Although the fuzzy system will make its decisions based on these rules, the system is not an abstruse "natural-language" artificial-intelligence system. Just like any other control system, the fuzzy system will take in numeric inputs from its sensors and output numeric data to control the traffic light.

Is a fuzzy system appropriate for this application? Or, more to the point, would you use a fuzzy approach for traffic metering outside this tutorial setting on a real freeway?

For several reasons, the answer is yes. Although an on-ramp metering system seems on the surface to be simple, the system is actually quite complex and, to a great degree, nonlinear. The behavior of human drivers tends to be nonlinear but also fairly predictable and, therefore, not random.

This nonlinear physical system is difficult to model mathematically, but you can easily express the sys-

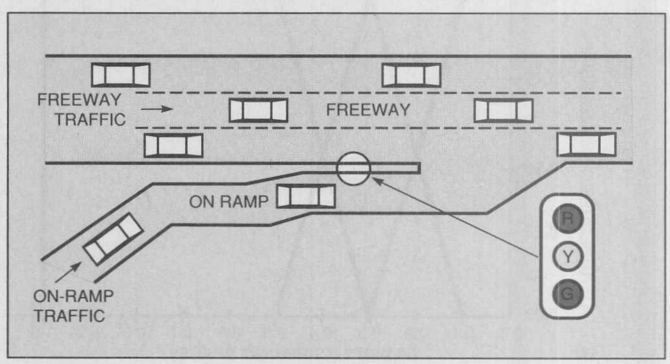


Fig 1—The fuzzy-logic system in this article will control the flow of cars from an on ramp onto a busy freeway by cycling a traffic light according to the freeway traffic's density and speed.

tem's operation in linguistic rules using fuzzy variables. You can also accomplish any needed subsequent modifications to the system quite simply.

System design

As is the case with designing many complex, nonmodeled systems, designing a fuzzy rule-based system is largely a seat-of-the-pants job. Recognizing the lack of a formal procedure, this example uses the following four steps:

1. Identify system inputs and those inputs' fuzzy ranges and establish degree-of-membership (or truth-value) functions for each range.
2. Identify outputs and those outputs' fuzzy ranges, again including the degree-of-membership functions.
3. Identify the rules that map the inputs to the outputs.
4. Determine the method of combining fuzzy rule actions into executable, "crisp" system outputs.

The system has two inputs: SPEED—the current average speed of the freeway traffic, and DENSITY—the current average density of the freeway traffic. In the real world these two inputs are somewhat tightly linked: Density tends to decrease as freeway speed increases because drivers allow greater separations between themselves and the car in front. However, we shall treat the two inputs separately.

Each input can assume a number of linguistic values, each value represented by a fuzzy set. To the fuzzy variable DENSITY we'll assign three fuzzy values (**Fig 2**): HEAVY—separation between cars is minimal, MEDIUM_HEAVY—separation between cars is nominal, and LIGHT—Separation between cars is maximum.

The fuzzy variable SPEED will also have three fuzzy values: SLOW—traffic is moving slowly, MEDIUM_FAST—traffic is moving at a nominal velocity below the speed limit, and FAST—traffic is moving at or above the speed limit.

We must deal finally with numbers—the linguistic values suggested have no quantitative meaning. **Fig 2** shows the degree-of-membership functions associated with the two inputs. Each linguistic fuzzy value has a corresponding fuzzy set. The shape of the fuzzy set determines its degree-of-membership, or "truth-value," function. Notice, for example, that the speed of 15 mph has a degree of membership of about 0.40 in both the SLOW and MEDIUM_FAST fuzzy sets. We arrived at the fuzzy sets in **Fig 2** arbitrarily, although an inherent requirement of a linguistically represented system is that the values be intuitively valid.

Identify system outputs

The system will have two outputs:

1. GREENLIGHT—The duration of the green-light state, in seconds, during which cars may enter onto the freeway.
2. REDLIGHT—The duration of the red-light state, also in seconds (which will include a constant-period yellow-light state), during which on-ramp cars may not enter onto the freeway.

Three fuzzy values apiece handle the two output variables nearly identically: SHORT—the given light is on only a short time, MEDIUM_LONG—the given light is on a medium period of time, LONG—the given light is left on for a long period of time. **Fig 3** shows the membership functions of these fuzzy values, as functions of time (seconds). In addition to these values, GREENLIGHT must have one additional (and crisp) value: CONSTANT_ON—the green light is on continuously.

After assigning input and output values to defined

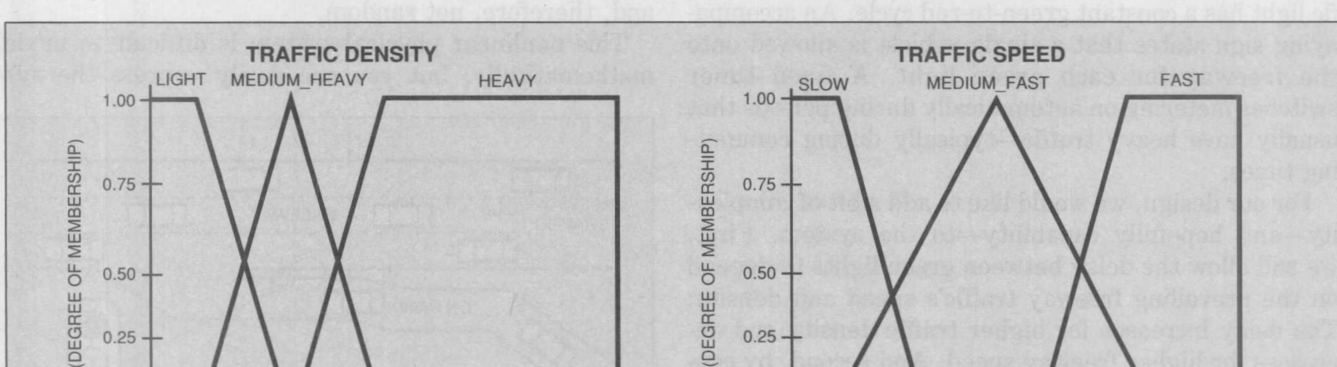


Fig 2—The degree-of-membership functions for the two system inputs, DENSITY (a) and SPEED (b), transform the measured (crisp) inputs into degrees of membership in fuzzy sets.

fuzzy sets, we must map each of the possible nine input conditions to an output. You do this mapping through the rule base. The most common form of expression for rules is linguistic:

if (condition or antecedent) then (action or consequence).

For example,

if (DENSITY is HEAVY and SPEED is SLOW) then (GREENLIGHT is SHORT, REDLIGHT is LONG).

You must keep in mind the heart of a fuzzy rule-based system: the *degree* to which the actions are executed corresponds directly to the *degree* to which the respective input conditions are true. The "AND" operator in the statement of the condition is a fuzzy AND, not a Boolean AND. The fuzzy AND means that the lesser value of the two degree-of-membership functions gets taken.

Continuing in this manner completes the rule base. A more compact form, and one that ensures we have covered all condition combinations, is the matrix representation. **Fig 4** shows both a list of rules and equivalent matrices for the metering-light rule base.

Several techniques exist for combining fuzzy actions into a single, crisp output. The fuzzy development and simulation program we used to verify this design supports only one: the centroid technique. The centroid technique, incidentally, is currently the most popular technique.

In making our first go at the design, we implemented the system using Hyperlogic's Cubicalc fuzzy tool and a simulator. The simulator provided a range of densities of simulated freeway traffic and several speed ranges for each density as inputs to the system. Resulting simulated outputs are the length of delay for each red light (that is, delay between green lights) and the number of cars allowed through for each green light.

To show how the fuzzy inference mechanism works, we'll step through a single iteration. The concept is not difficult, but the process has a number of potentially confusing steps. Refer to **Figs 5** and **6**.

We assume a traffic density of 0.35 (which trans-

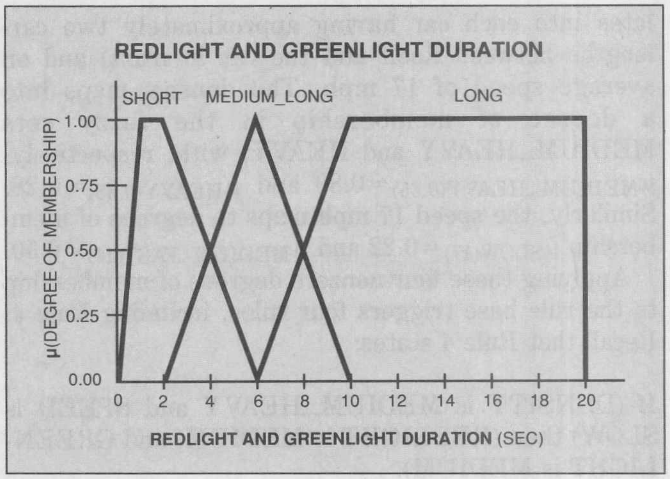


Fig 3—The degree-of-membership functions for the two system outputs, REDLIGHT and GREENLIGHT, transform the fuzzy actions that the rule base specifies back into executable (crisp) outputs.

1 if (DENSITY is HEAVY and SPEED is SLOW) then (REDLIGHT is LONG, GREENLIGHT is SHORT)
2 if (DENSITY is HEAVY and SPEED is MEDIUM_FAST) then (REDLIGHT is MEDIUM_LONG, GREENLIGHT is MEDIUM_LONG)
3 if (DENSITY is HEAVY and SPEED is FAST) then (REDLIGHT is SHORT, GREENLIGHT is LONG)
4 if (DENSITY is MEDIUM_HEAVY and SPEED is SLOW) then (REDLIGHT is MEDIUM_LONG, GREENLIGHT is MEDIUM_LONG)
5 if (DENSITY is MEDIUM_HEAVY and SPEED is MEDIUM_FAST) then (REDLIGHT is SHORT, GREENLIGHT is LONG)
6 if (DENSITY is MEDIUM_HEAVY and SPEED is FAST) then (REDLIGHT is (SHORT), GREENLIGHT is CONSTANT_ON)
7 if (DENSITY is LIGHT and SPEED is SLOW) then (REDLIGHT is SHORT, GREENLIGHT is LONG)
8 if (DENSITY is LIGHT and SPEED is MEDIUM_FAST) then (REDLIGHT is (SHORT), GREENLIGHT is CONSTANT_ON)
9 if (DENSITY is LIGHT and SPEED is FAST) then (REDLIGHT is (SHORT), GREENLIGHT is CONSTANT_ON)

(a)

REDLIGHT

DENSITY \ SPEED	SLOW	MEDIUM_FAST	FAST
LIGHT	7 (SHORT)	8 (SHORT)	9 (SHORT)
MEDIUM_HEAVY	4 MEDIUM_LONG	5 SHORT	6 (SHORT)
HEAVY	1 LONG	2 MEDIUM_LONG	3 SHORT

GREENLIGHT

DENSITY \ SPEED	SLOW	MEDIUM_FAST	FAST
LIGHT	7 CONSTANT_ON	8 CONSTANT_ON	9 CONSTANT_ON
MEDIUM_HEAVY	4 MEDIUM_LONG	5 LONG	6 CONSTANT_ON
HEAVY	1 SHORT	2 MEDIUM_LONG	3 LONG

(b)

Fig 4—You can represent the rules that relate inputs to outputs in linguistic form (a) or in matrix form (b). The rules map the various fuzzy values of the inputs DENSITY and SPEED to the outputs GREENLIGHT and REDLIGHT. Note that four SHORT values of REDLIGHT have parentheses. The parentheses indicate a don't-care state because the corresponding state for GREENLIGHT is CONSTANT_ON.

lates into each car having approximately two car-lengths between itself and the car in front) and an average speed of 17 mph. The density maps into a degree of membership in the fuzzy sets MEDIUM_HEAVY and HEAVY, with, respectively, $\mu_{\text{MEDIUM\_HEAVY}(0.35)}=0.80$ and $\mu_{\text{HEAVY}(0.35)}=0.28$. Similarly, the speed 17 mph maps to degrees of membership $\mu_{\text{SLOW}(17)}=0.22$ and $\mu_{\text{MEDIUM\_FAST}(17)}=0.50$.

Applying these four nonzero degrees of membership to the rule base triggers four rules, including Rule 4. Recall that Rule 4 states:

If (DENSITY is MEDIUM_HEAVY and SPEED is SLOW) then (REDLIGHT is MEDIUM, and GREENLIGHT is MEDIUM).

We have already determined that DENSITY is MEDIUM_HEAVY ($\mu=0.80$) and SPEED is SLOW ($\mu=0.22$). The fuzzy AND of these two expressions is the lesser of the two values, so the truth value (which, again, for our purposes is synonymous with "degree of membership") of the input condition for Rule 4 is $\mu_{\text{CONDITION}}=0.22$.

This result we apply to both output actions:

$$\mu_{\text{MEDIUM\_LONG(REDLIGHT)}} = \mu_{\text{MEDIUM\_LONG(GREENLIGHT)}} = 0.22.$$

Had Rule 4 been the only rule that fired, these values would also uniquely determine the actual executed values for REDLIGHT and GREENLIGHT. However, given the two specific inputs, a total of four rules will fire: Rules 1, 2, 4, and 5. By following the same steps as we did for Rule 4, we would find that the calculated REDLIGHT and GREENLIGHT durations are (here including the Rule 4 results)

$$\mu_{\text{LONG(REDLIGHT)}} = \mu_{\text{SHORT(GREENLIGHT)}} = 0.22 \qquad \textbf{(Rule 1)}$$

$$\mu_{\text{MEDIUM\_LONG(REDLIGHT)}} = \mu_{\text{MEDIUM\_LONG(GREENLIGHT)}} = 0.28 \qquad \textbf{(Rule 2)}$$

$$\mu_{\text{MEDIUM\_LONG(REDLIGHT)}} = \mu_{\text{MEDIUM\_LONG(GREENLIGHT)}} = 0.22 \qquad \textbf{(Rule 4)}$$

$$\mu_{\text{SHORT(REDLIGHT)}} = \mu_{\text{LONG(GREENLIGHT)}} = 0.50. \qquad \textbf{(Rule 5)}$$

We must now combine and translate these results into a crisp, executable output. As **Fig 6** shows, we use the centroid technique to do so for the REDLIGHT duration. Again using Rule 4's action, the membership function of MEDIUM_LONG is sliced off at the designated truth value, $\mu=0.22$. The centroid of the resulting area is at $\text{REDLIGHT}_{\text{Rule 4}}=6$ sec.

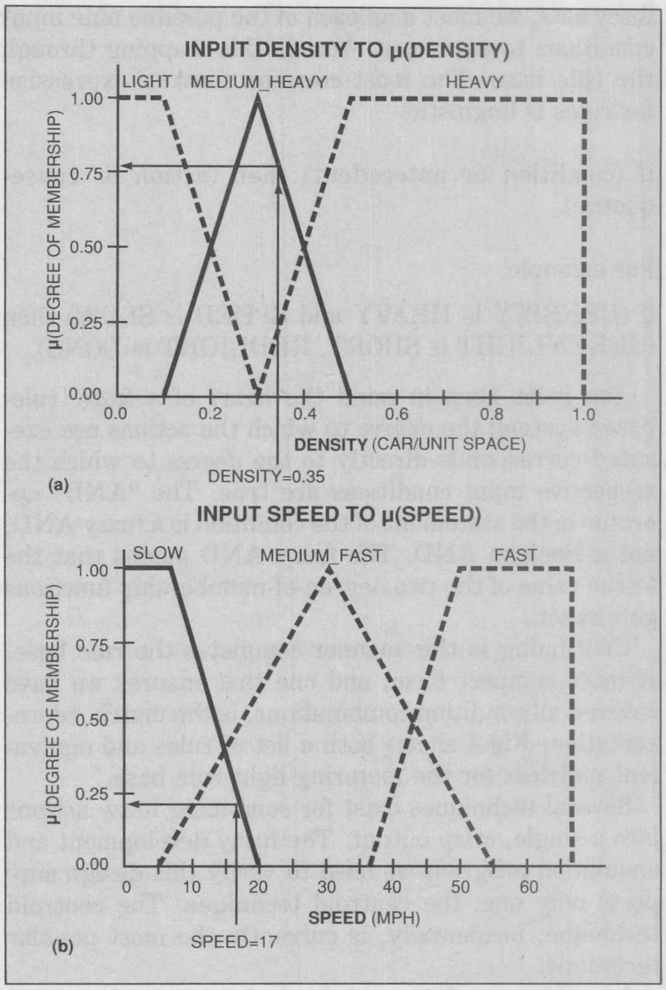


Fig 5—A traffic density of 0.35 translates into $\mu=0.80$ in the fuzzy set MEDIUM_HEAVY (a). Similarly, a traffic speed of 17 mph yields $\mu=0.22$ in the fuzzy set SLOW (b).

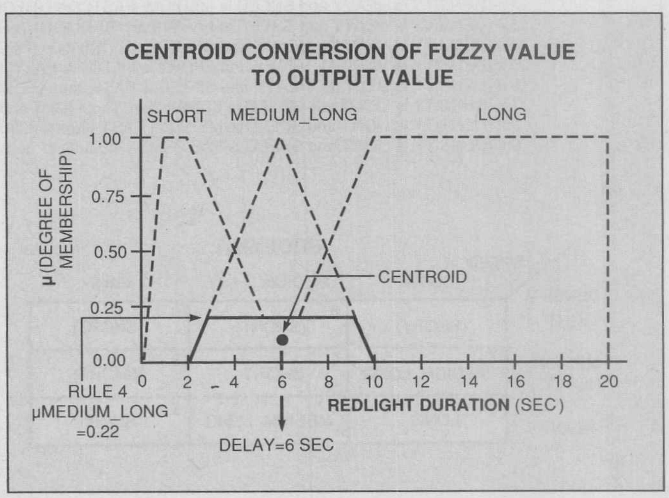


Fig 6—The centroid method of combination/defuzzification crops the fuzzy set MEDIUM_LONG at a height of 0.22—the result from Fig 5. The centroid of the resultant truncated figure is at a delay of 6 seconds.

In a similar manner, calculating the centroids for the output areas corresponding to the actions of Rules 1, 2, and 5 yields: $REDLIGHT_{Rule\ 1}=13.2$ sec, $REDLIGHT_{Rule\ 2}=6$ sec, and $REDLIGHT_{Rule\ 5}=2.75$ sec. A weighted-average method, where the weights are the respective truth values, combines all these centroids. The final output is:

$$t_{REDLIGHT}=((13.2)(0.22)+(6)(0.28)+(6)(0.22)+(2.75)(0.50))/(0.22+0.28+0.22+0.50)=5.96 \text{ seconds.}$$

Performance is pretty much as expected. For light traffic, the system allows cars onto the freeway with little or no delay and with little impact on freeway speed and density. As the number of cars on the freeway (DENSITY) increases, the number of cars allowed in from the on ramp decreases dramatically.

Obviously, the design suffers from a significant oversight. This oversight is an intentional setup; it allows for an extra modification step, and the ease of modification of fuzzy systems is important. When traffic is very dense, the system allows a bare minimum of cars, approaching zero, to enter from the on ramp. With a little thought, the reason becomes obvious. If the design goal, implicit in the rule base, is to optimize freeway density, then freeway density is least adversely affected when the system allows no cars to enter from the on ramp. Thus, while the design goal of optimizing freeway traffic flow is desirable to those already on the freeway, it would not be a popular one among the drivers stacking up on the on ramp.

System modification

We need to modify our system. The system as it stands now has a natural correctional feedback (drivers wanting to enter onto the freeway that see an overly long line at the on ramp will tend to look for an alternate route), but it is minimally effective. To improve on the design, we must include another input and a corresponding additional set of rules to balance the "minimal impact on freeway density" goal with a "minimal number of cars on the on ramp" goal.

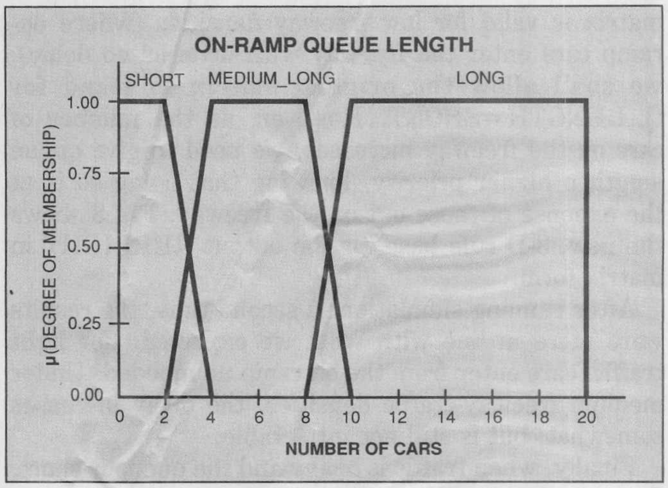


Fig 7—Simulations reveal that the rules in Fig 4 will allow no cars onto the freeway if the prevailing traffic is heavy. Having added these Q_LENGTH fuzzy sets and modified the rule base, the system will now attempt to balance the need to optimize freeway-traffic density with the need to minimize on-ramp queue length.

The input we'll add is the number of cars currently on the ramp waiting to be metered onto the freeway—call it Q_LENGTH. As with the other inputs, Q_LENGTH is a fuzzy variable and comprises three fuzzy values, each with its own degree of membership function: SHORT—a few cars are on the on ramp, MEDIUM_LONG—a moderate number of cars are on the on ramp, and LONG—many cars are on the on ramp. **Fig 7** shows these functions.

Adding Q_LENGTH to the rule base adds a third dimension to the original matrix. Because the original

Fuzzy logic evinces interest from US engineers

Even a year ago, fuzzy-logic practitioners, recognizing the highly publicized successes in Japan, were bemoaning US companies' utter lack of interest in fuzzy-logic systems. The Western engineering community, caught up in the need for precision that has been evident as far back as Aristotle, was incapable of appreciating—much less applying—fuzzy logic. The term "fuzzy" itself was an encumbrance, part of an overall psychological barrier that embodied a distrust of anything imprecise.

This situation is changing rapidly. The Japanese still dominate in fuzzy applications, but American engineers are now starting to catch on, often through the efforts of internal individual and small-group "champions" that learned the technology, caught the spirit, and are now busy converting their colleagues and management.

The psychological barrier is coming down. And if we can keep from getting caught up in a self-defeating hype campaign, we will come to accept fuzzy logic as a viable and powerful system-design paradigm.

The bottom line is that, because fuzzy logic is a fundamental mathematical technique, there are few disciplines where you cannot apply it.

matrix is valid for low freeway densities (where on-ramp cars enter the freeway with little or no delay), we shall allow the original matrix to stand for Q_LENGTH=SHORT. However, as the number of cars on the freeway increases, we need to give queue length a higher priority, knowing that doing so is at the expense of those out on the freeway. **Fig 8** shows the new, 3-D rule base for the output REDLIGHT in matrix form.

After running simulations a second time, the results were more in line with what we expected. For light traffic, cars enter from the on ramp unimpeded. Under medium freeway-traffic densities, the delay increases somewhat, but is still not intolerable.

Finally, when traffic is heavy and the queue is short, freeway density dominates and on-ramp delays are long; for increasing queue lengths, the need to move cars onto the freeway becomes more important, and freeway density suffers. We now have a system whose operation is consistent with what we would expect to see were this system used in an actual application.

Why did we simulate the system? Currently, rule-based fuzzy systems are nonanalytic. By expanding fuzzy systems' capability to handle extremely complex and nonlinear systems, we have sacrificed the ability to analyze mathematically their correct operation and complete representation of the system being controlled. To ensure correct operation (or at least an increased comfort margin) we must simulate the system's operation. This nonanalytic approach often rankles those who are grounded in traditional, linear theory but, if done correctly, it is tested and reliable.

Porting to the real world

Several options exist to assist porting a fuzzy rule-based system to a real-world application. The first option is to develop your own fuzzy system. As you can see from the example, the basic architecture has three stages:

1. A crisp-to-fuzzy transformation of inputs.
2. Applying these fuzzy inputs as conditions to the rule base.
3. Combining the resulting actions and transforming from a fuzzy set of outputs back to executable, crisp outputs.

Actually, although developing your own system may sound involved, it is not all that difficult. The real complexity in developing a fuzzy system is in creating and testing both the degree-of-membership functions and the rule base, rather than in implementing the runtime environment.

The second porting option is to use an embedded architecture that supports DOS and utilize the runtime environments provided by the various toolmakers. Finally, both Hyperlogic and Togai Infralogic optionally provide the C source code they use in their runtime environment, which will allow cross compilation into nearly any target.

The example is conceptually straightforward. The intent of this article is not to design a traffic light but rather to demonstrate how to design a fuzzy system. You can follow the approach outlined here to develop even relatively complex fuzzy rule-based systems.

Note that at no time did we have to create a model of the physical system. The solution used linguistically stated rules describing actions that are in response to inputs. This lack of a model is the dominant strength

Q_ LENGTH is SHORT

DENSITY \ SPEED	SLOW	MEDIUM_FAST	FAST
LIGHT	(SHORT)	(SHORT)	(SHORT)
MEDIUM_ HEAVY	MEDIUM_LONG	SHORT	(SHORT)
HEAVY	LONG	MEDIUM_LONG	SHORT

Q_ LENGTH is MEDIUM_LONG

DENSITY \ SPEED	SLOW	MEDIUM_FAST	FAST
LIGHT	(SHORT)	(SHORT)	(SHORT)
MEDIUM_ HEAVY	SHORT	SHORT	(SHORT)
HEAVY	MEDIUM_LONG	SHORT	(SHORT)

Q_ LENGTH is LONG

DENSITY \ SPEED	SLOW	MEDIUM_FAST	FAST
LIGHT	(SHORT)	(SHORT)	(SHORT)
MEDIUM_ HEAVY	(SHORT)	(SHORT)	(SHORT)
HEAVY	SHORT	(SHORT)	(SHORT)

Fig 8—Compare these three matrices for REDLIGHT with the single REDLIGHT matrix in Fig 4. When the queue's length is short, the rules are the same as Fig 4's. But as the queue gets longer, the rules make the REDLIGHT interval shorter. (The GREENLIGHT interval, not shown, is the complement of the REDLIGHT interval.)

FLUKE®

PHILIPS

Test results:

90% of those who try a Philips Logic Analyzer from Fluke buy one.

100% get a free DMM*.

Our logic analyzers sell themselves. All we have to do is get one in your hands. To make sure you do, we're giving you a Fluke DMM*, whether you buy our analyzer or the competition's.

Only the Philips PM 3580 family of logic analyzers give you *true* dual state and timing on up to 96 channels - simultaneously. All accessible with one probe and one keystroke. Which means no more dual probing or reconfiguration between state and timing. Or no probes at all if you use our boundary-scan test option!

*The top-of-the-line Fluke 12 in our newest DMM family. It combines a smart set of troubleshooting features in a new design that's exceptionally fast and simple to operate – with one hand. It's yours after our 30 minute demo, no matter whose logic analyzer you purchase. Offer expires September 30, 1992.

All our analyzers feature 50 MHz state and up to 200 MHz timing speeds. As well as integrated state and timing triggering for fast debug of complex hardware and software problems. Plus broad µp support like Intel®'s i486; i386; 80286; 80186/88 families. The MCS-96, 8051, and i960 families. And the Motorola 68040 to 6800, 68HC11, 68332/1, 68302, 68340, 56001, AMD®'s AM 29030, and TI's 320Cxx family.

The PM 3580 family of logic analyzers is priced from $4495 to $11,450 - about half the cost of comparable analyzers. What's more you can have them up and running in only 30 minutes.

Find out why the PM 3580 family of logic analyzers were the only ones cited for excellence and innovation by *Electronic Design, EDN, Embedded Systems, Electronic Products*, and *R&D* magazines. Take the Fluke Challenge. The odds are 100% you'll be totally impressed.

For literature, our video or a demonstration, call **1-800-44-FLUKE**.

John Fluke Mfg. Co., Inc., P.O. Box 9090, M/S 250C, Everett, WA 98206-9090. U.S. (206) 356-5400. Canada (416) 890-7600. Other countries: (206) 356-5500.

FAST ANSWERS

FLUKE®

CIRCLE NO. 74

of a fuzzy rule-based system. Using intuitive terms rather than abstract mathematical models, you can implement complex systems rapidly.

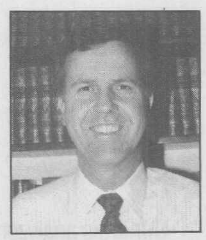
We have presented a simple fuzzy system. It demonstrates the lowest level of how to incorporate fuzzy logic into system design. Higher-level fuzzy structures can take the form of either adapting traditional, crisp-level structures or creating new structures that are achievable only with fuzzy logic.

A fuzzy solution may be the only one possible. For example, fuzzy decision and scheduling schemes handle imprecise inputs, logically combining them to provide some optimal decision or schedule. In addition, fuzzy evidence techniques allow collecting both deductive and inductive evidence in forming conclusions, again in light of incomplete or uncertain data.

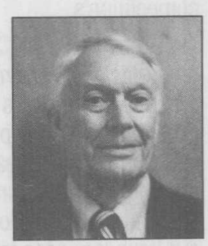
One final point. Proponents often claim that fuzzy logic sacrifices precision for a more accurate "big-picture" view of the system. This is not entirely correct. Fuzzy logic does retain the high-level view of a system better, but it also only sacrifices *unnecessary* precision. If you need greater precision, you can get it from your fuzzy system by, for example, increasing the number of input and output values and accounting for input-to-output mapping through an increased number of more specific rules. EDN

To get a foundation in the basics of fuzzy logic, turn to pg 111 in this issue.

Authors' biographies



David Brubaker is president of The Huntington Group (Menlo Park, CA), consultants in the design of systems using real-time embedded processors and fuzzy logic. He has worked with Sun Microsystems, Beckman Instruments, Motorola, TRW, Ford, and ESL. David holds BS, MS, and PhD degrees, all in electrical engineering, from Stanford University. His personal activities include walking, backpacking, and coaching his children's basketball teams.



Cedric Sheerer is president of C/S Associates (Los Altos, CA). He holds degrees in both electrical engineering and business and has 23 years of experience as a software and hardware engineer and an international consultant. His professional specialties include artificial intelligence (AI), video-digital imaging, data acquisition, and algorithmic signal processing, which he has used to develop AI-based analog/digital board-level diagnostic tools.

Article Interest Quotient (Circle One)
High 476 Medium 477 Low 478